Introduction to the OCplus package

Alexander Ploner
Medical Epidemiology & Biostatistics
Karolinska Institutet, Stockholm
email: alexander.ploner@ki.se

May 3, 2016

1 Overview

The `OCplus` package offers a variety of tools for designing and analyzing gene expression microarray experiments. The common underlying statistical concept is the use of the false discovery rate (fdr) to identify differentially expressed (DE) genes.

A commonly underappreciated fact is that in a microarray setting, magical thresholds like 0.05 or 0.01 make even less sense for the fdr than they do for the traditional p-values. The trade-off between fdr and the ability to detect relevant genes can be made much more explicit than the classical trade-off between p-value and statistical power. A central idea of `OCplus` is to allow the user to make up her mind about the trade-off appropriate for her specific situation, based on the operating characteristics of her experimental design or data set (hence the name).

The main functionality of `OCplus` falls into three categories, listed below with their most important functions:

1. Sample size assessment: `TOC`, `samplesize`
2. Data analysis: `EOC`, `fdr1d`, `fdr2d`
3. Estimation of the proportion of non-DE genes: `tMixture`

This short introduction explains the underlying model and demonstrates the main functionality in each category; in-depth descriptions can be found in the individual vignettes.

2 Installation

You need the package `akima`, available from CRAN. In order to run `EOC`, you also need the package `multtest` from Bioconductor.

```
> library(OCplus)
```

3 Sample size calculations

3.1 samplesize

This function allows the user to choose an appropriate number of microarray chips per group for an assumed proportion of regulated genes with a minimum fold change. Specifically, the function calculates the global false discovery rate (FDR) among genes with the absolute largest t-statistics, assuming a given proportion `p0` of non-differentially expressed (nonDE) genes, and a given effect size `D` for the differentially expressed (DE)genes:

```
> ss1 = samplesize(p0=0.95, D=1, crit=0.01)
```

In the example above, we assume that 95% of all genes are nonDE, and that the 5% DE genes have a log2-fold change of `D`= ± 1 (i.e. a fold change of 0.5 and 2, respectively); this produces the following result:

```
> ss1

       FDR_0.01     fdr_0.01
5  6.383069e-01 0.6993141173
10 2.525235e-01 0.3799927428
15 7.292893e-02 0.1467379021
20 1.793485e-02 0.0430765448
25 4.272366e-03 0.0114247637
30 1.033032e-03 0.0029846590
35 2.555826e-04 0.0007864970
40 6.458336e-05 0.0002098690
45 1.661407e-05 0.0000566670
50 4.338653e-06 0.0000154595
```

The listed FDRs are for the genes with 1% largest t-statistics (or equivalently, the 1% smallest p-values). We find that for $n = 5$ microarray chips per group, these genes have a FDR of 64%, meaning that roughly 2/3 of the top genes can be expected to be false positives; if we invest however in $n = 5$ microarray chips per group, less than 2% of the top genes will be false positives.

As a a side effect, `samplesize` produces a plot of the FDR as a function of the number of chips per group, as shown in Figure **??**. It shows that there is little to gain by increasing group sizes beyond $n = 20$.

3.2 TOC

This function calculates the theoretical operating characteristics of a chosen design; for a given group size, proportion of regulated genes and minimum fold changes, the function shows the trade off between FDR and sensitivity for any possible threshold on the t-statistics.

```
> samplesize(p0=0.95, D=1, crit=0.01)
```

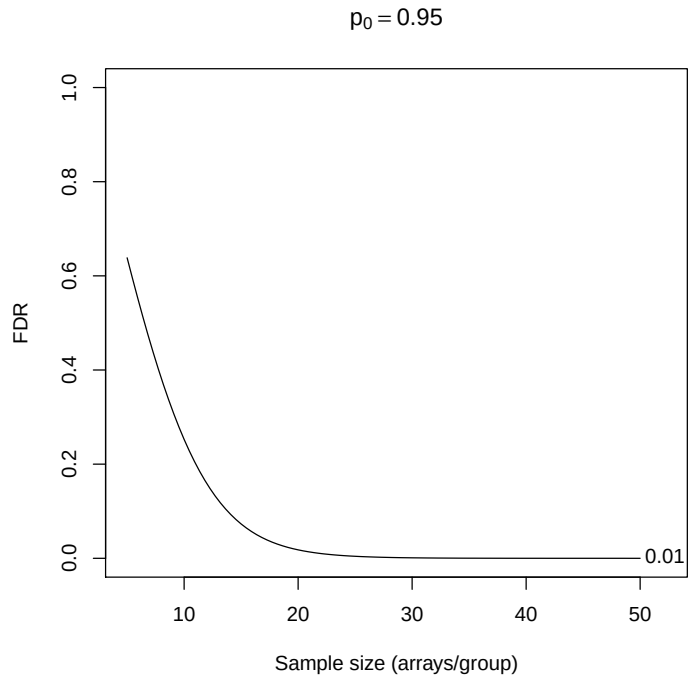

Figure 1: FDR as a function of samplesize, assuming that genes with 1% absolutely largest t-statistics are declared DE.

```
> TOC(n=20, p0=0.95, D=1, alpha=FALSE, legend=TRUE)
```

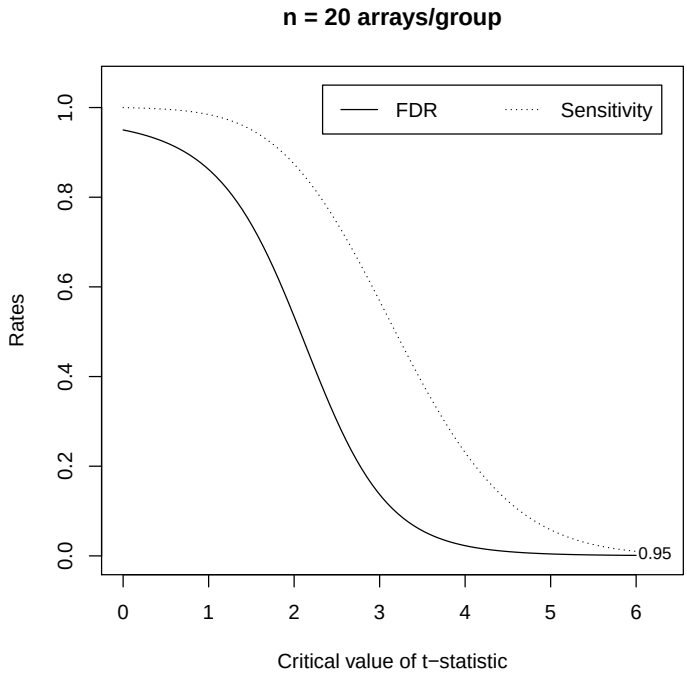

Figure 2: FDR and sensitivity as a function of the threshold for declaring a gene to be DE

4 Identifying DE genes

`OCplus` offers three different functions for identifying differentially expressed genes. All three are based on different variants of the false discovery rate: `EOC` computes the global false discovery rate (FDR) for each gene, `fdr1d` and `fdr2d` compute different variants of the local false discovery rate (fdr). Using the FDR is the conventional and most direct approach and works generally out of the box. The fdr approach is potentially more powerful, because it uses smoothing to combine information across genes, but it may require some experimentation to get the smoothing paramters right: `fdr1d` will often work with the default settings, but `fdr2d` will usually require some modifications (but is proportionally more powerful than `fdr1d`).

We use (unrealistically simple, but convenient) simulated data in the following to demonstrate these approaches:

```
> set.seed(123)
> simdat = MAsim(ng=10000, n=10, p0=0.95, D=1, sigma=1)
> dim(simdat)

[1] 10000    20

> colnames(simdat)

 [1] "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "1" "1" "1" "1" "1"
[16] "1" "1" "1" "1" "1"
```

`simdat` contains the simulated log-expression values for 10,000 genes and two groups of samples with 10 chips per group; the log-expression values are assumed normal and independent, with standard deviation one and mean zero for the 95% non-DE expressed genes, and mean ± 1 for the DE genes in the second group.

4.1 EOC

This function is the counterpart to `TOC` and returns the empirical operating characteristics: for each gene, the associated t-statistic, p-value, FDR and sensitivity.

```
> sim1 = EOC(simdat, colnames(simdat))
> sim1[1:5,]

        tstat   pvalue       FDR      sens
1 -1.08042708 0.294052 0.8833315 0.9522985
2 -0.01011066 0.992172 0.9578601 1.0000000
3  1.07476286 0.296772 0.8852288 0.9522985
4  1.55454200 0.136740 0.7909227 0.8506491
5 -1.53863046 0.140456 0.7958018 0.8506491
```

Note that this function plots the operating characteristics by default, but this can be suppressed by setting the argument `plot=FALSE`, and the output still has its own plotting method, see Figure **??**.

The genes with the smallest FDR can be extracted via `topDE`:

```
> topDE(sim1, co=0.1)
```

```
         tstat   pvalue        FDR        sens
2418 -6.624755 0.000000 0.00000000 0.002445655
7286 -6.923683 0.000000 0.00000000 0.000000000
1357  5.485208 0.000028 0.03836445 0.014032012
3480 -5.762388 0.000016 0.03836445 0.004534915
7589 -5.697200 0.000020 0.03836445 0.006917298
8399 -5.447536 0.000032 0.03836445 0.016407036
8432  5.583280 0.000028 0.03836445 0.011656393
8767  5.595279 0.000028 0.03836445 0.009219307
9195  5.398711 0.000040 0.04262716 0.018687317
261   5.232498 0.000052 0.04475852 0.023299550
8115 -5.176199 0.000056 0.04475852 0.025660413
9207  5.239546 0.000052 0.04475852 0.020864015
324  -5.100353 0.000072 0.05312000 0.027749115
1804 -5.061102 0.000080 0.05480635 0.029943787
3934 -4.966195 0.000104 0.06330134 0.031871388
4116  4.921912 0.000112 0.06330134 0.034127753
4267 -4.896086 0.000128 0.06330134 0.041073370
4951  4.920252 0.000112 0.06330134 0.036547935
6108 -4.899887 0.000128 0.06330134 0.038648214
6870 -4.864438 0.000132 0.06330134 0.043389642
1150 -4.788900 0.000160 0.07149738 0.045201889
3687  4.777048 0.000164 0.07149738 0.047561365
7683 -4.737323 0.000180 0.07506087 0.049587579
526   4.658298 0.000212 0.08299248 0.053740760
632   4.658329 0.000212 0.08299248 0.051296297
515   4.584209 0.000248 0.08951704 0.055397898
7731 -4.564034 0.000252 0.08951704 0.057693452
2282  4.506643 0.000288 0.09865143 0.059360367
```

The proportion of non-DE genes `p0` is by default estimated from the data using a variant of Storey's method; the estimate can be extracted from the output:

```
> p0(sim1)
```

```
[1] 0.9591112
```

`p0` can also be specified explicitly in the function call, if an alternative estimate is availabale, see Section **??**.

```
> plot(sim1)
```

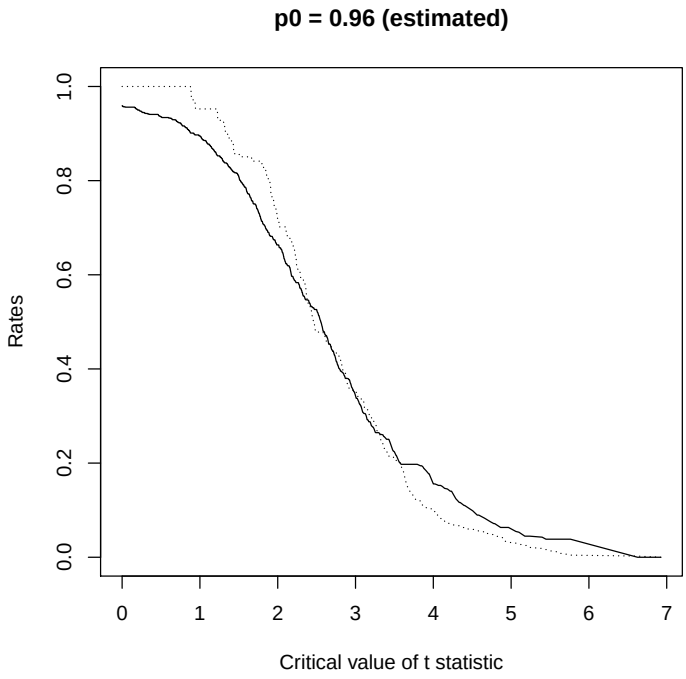

Figure 3: Estimated FDR and sensitivity for the simulated data

4.2 fdr1d

This function returns for each gene the test statistic and the local (univariate) fdr:

```
> sim2 = fdr1d(simdat, colnames(simdat), verb=FALSE)
> sim2[1:5,]

        tstat fdr.local
1  1.08042708 0.9919021
2  0.01011066 0.9816645
3 -1.07476286 0.9371344
4 -1.55454200 0.9254457
5  1.53863046 0.9609410
```

The `verb=FALSE` here just stops the function from reporting the number of the current permutation, which creates too much output for a vignette. `fdr1d` does not plot automatically, but has its own plotting method, see Figure **??**.

The proportion of non-DE genes `p0` is by default estimated from the data, using a variant of Efron's method. The estimate can again be extracted from the output via the function `p0`; it is also reported by the summary method:

```
> summary(sim2)

$Statistic
    Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
-5.59500 -0.74860 -0.01244 -0.01883  0.68860  6.92400

$fdr
         fdr
statistic (0,0.05] (0.05,0.1] (0.1,0.2] (0.2,1] (1,Inf]
     t<0         2          3        10    4941       0
     t>=0        1          5        14    5024       0

$p0
$p0$Value
[1] 0.9356216

$p0$Estimated
[1] TRUE
```

Note that the value for `p0` is very close to the estimate used in `sim1` above.

The genes with fdr below a specified threshold can again be listed by

```
> topDE(sim2, co=0.1)

         tstat  fdr.local
7286  6.923683 0.01930930
```

```
> plot(sim2)
```

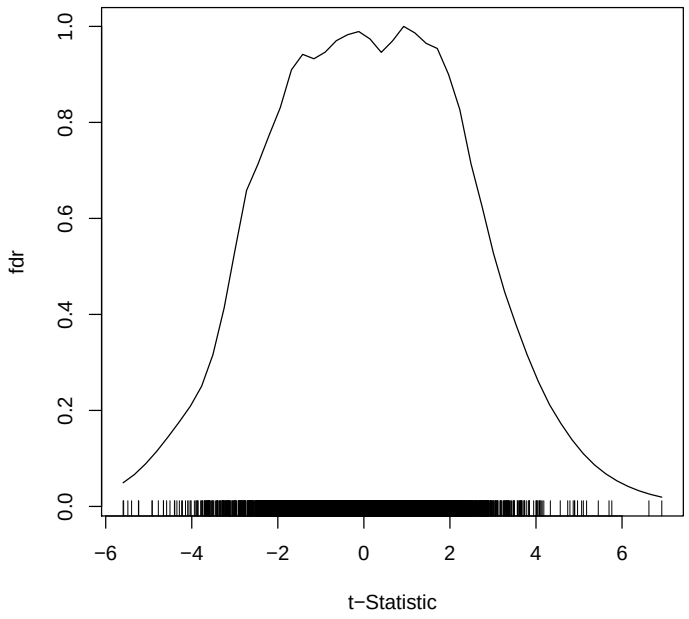

Figure 4: Local fdr as a function of t-statistics for the simulated data; inner ticks on the horizontal axis indicate observed t-statistics.

```
2418  6.624755 0.02607632
8767 -5.595279 0.04928532
8432 -5.583280 0.05006909
1357 -5.485208 0.05647513
3480  5.762388 0.06001107
9195 -5.398711 0.06212512
7589  5.697200 0.06368385
9207 -5.239546 0.07445790
261  -5.232498 0.07506200
8399  5.447536 0.08027630
```

4.3 fdr2d

This function reports for each the test statistic, the auxiliary test statistic (generally the log of the standard error) and the local fdr based on the two test

statistics:

```
> sim3 = fdr2d(simdat, colnames(simdat), p0=p0(sim2), verb=FALSE)
> sim3[1:5,]

        tstat      logse fdr.local
1  1.08042708 -0.6146155 0.9003866
2  0.01011066 -1.0317545 0.9120523
3 -1.07476286 -0.7420905 0.8882119
4 -1.55454200 -1.0702954 0.8693983
5  1.53863046 -0.9119842 0.8854507
```

Note that here the proportion `p0` is specified explicitly – we take the estimate based on the univariate densities used for `sim2`. The default behavior for `fdr2d` is also to estimate `p0` from the data, but the results can be highly erratic, and we recommend using an external estimate, either from `EOC` or `fdr1d` as above, or from `tMixture`, as described in Section **??**.

As mentioned above, the smoothing parameter `smooth` of `fdr2d` will often require adjustment. A useful graphical diagnostic for a suitable value of `smooth` is shown in Figure **??**: in theory, the onedimensional fdr is equal to the twodimensional fdr averaged across the log standard errors; in Figure **??**, the solid line shows the onedimensional fdr (`sim2`) and the broken line shows the averaged twodimensional fdr; the agreement between the two lines is good, though better for low fdr (in the tail) than for high fdr (in the center). In practice, it is quite hard to achieve perfect agreement throughout, as different degrees of smoothing might be required in the center compared to the tails. We are, however, generally only interested in the genes with low fdr anyway, so it is usually sufficient to achieve a good fit in the tails.

The results can be summarized as above, and the top list extracted with the same method as for `fdr1d`:

```
> summary(sim3)

$Statistic
    Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
-5.59500 -0.74860 -0.01244 -0.01883  0.68860  6.92400

$fdr
          fdr
statistic (0,0.05] (0.05,0.1] (0.1,0.2] (0.2,1] (1,Inf]
     t<0        10         11        14    4809     112
     t>=0        8         11        20    4743     262

$p0
$p0$Value
[1] 0.9356216
```

```
> plot(sim2)
> lines(average.fdr(sim3), lty=2)
```

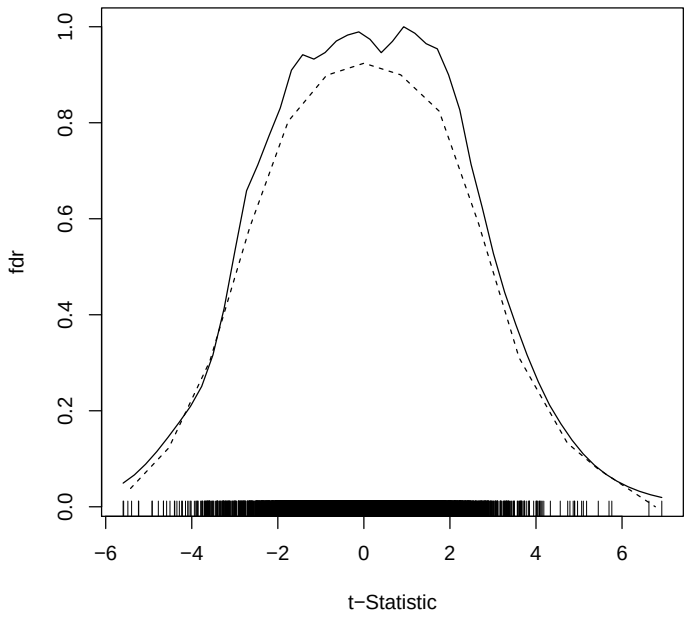

Figure 5: Local onedimensional fdr as a function of the t-statistic (solid line, as in Figure **??**) and averaged twodimensional fdr (broken line).

```
$p0$Estimated
[1] FALSE
```

```
> topDE(sim3, co=0.1)
```

```
         tstat      logse    fdr.local
2418  6.624755 -1.1640917 3.918071e-06
7286  6.923683 -1.1099516 3.918071e-06
8432 -5.583280 -1.0391987 3.918071e-06
8767 -5.595279 -1.0040761 3.918071e-06
6108  4.899887 -0.8199713 1.497317e-02
7589  5.697200 -1.1382646 1.657668e-02
6870  4.864438 -0.8269019 1.697618e-02
1357 -5.485208 -0.9759968 1.818932e-02
3480  5.762388 -1.1545476 2.130251e-02
3739 -4.345276 -0.7770245 2.340007e-02
9195 -5.398711 -0.9326397 2.555046e-02
3934  4.966195 -0.9017693 2.786776e-02
4243 -4.217978 -0.7644934 3.150729e-02
2612  4.044563 -0.6664508 3.455523e-02
8399  5.447536 -1.0913539 3.550615e-02
526  -4.658298 -0.8758258 4.196764e-02
1150  4.788900 -0.9348992 4.889755e-02
4116 -4.921912 -0.9357994 4.950849e-02
2946 -3.880855 -0.6957500 5.319847e-02
632  -4.658329 -0.9241577 5.896822e-02
6711  4.016070 -0.7679300 5.992003e-02
6400  3.739040 -0.7069355 6.467170e-02
1804  5.061102 -1.0147315 6.875727e-02
3425  3.840764 -0.7495248 6.916033e-02
1471 -3.706950 -0.7858991 6.936155e-02
5901 -3.688175 -0.7389362 6.995866e-02
5417  3.834645 -0.7790989 7.760831e-02
9085  3.383585 -0.5761734 7.764630e-02
2480  4.181968 -0.8827882 8.043395e-02
9737 -3.622699 -0.6799167 8.548070e-02
404   4.080794 -0.8681257 8.585259e-02
7257 -3.655494 -0.7834283 8.742277e-02
515  -4.584209 -0.9842774 8.967922e-02
324   5.100353 -1.0505343 9.019165e-02
9207 -5.239546 -1.1186978 9.061218e-02
3514 -3.635134 -0.7607702 9.332590e-02
261  -5.232498 -1.1232774 9.496853e-02
7683  4.737323 -1.0125250 9.528572e-02
6562 -3.878664 -0.8230592 9.568517e-02
9024  3.638140 -0.7548439 9.845330e-02
```

```
> plot(sim3)
```

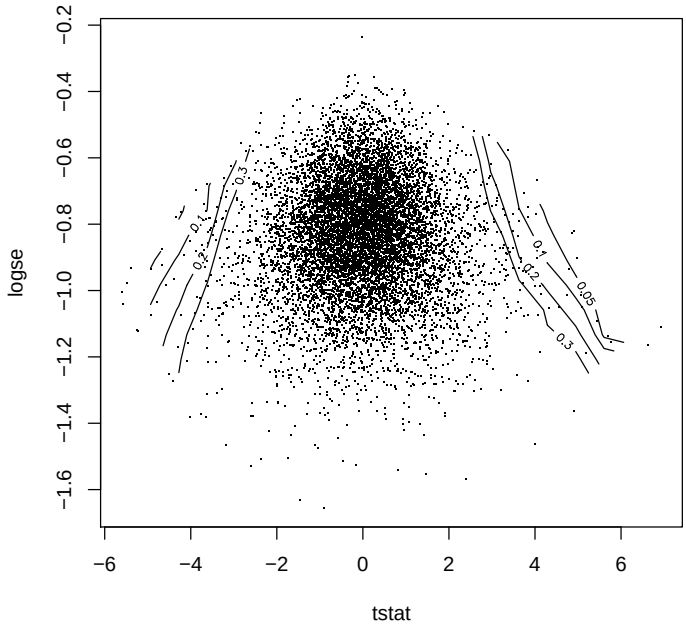

Figure 6: A scatterplot of the log standard errors vs. the t-statistics, with the estimated fdr indicated by isolines.

Note that the table of fdrs (`$fdr`) in this output contains fdrs greater than one; this, too, is a consequence of not quite correct smoothing for genes with large fdr.

Figure **??** shows the standard plot for output from `fdr2d`. This is basically a scatterplot of the two contributing statistics, with the estimated fdr overlayed as isolines. Note that the averaged values shown as a broken line in Figure **??** are calculated by averaging along the vertical axis (the log standard errors) in Figure **??**.

4.4 Compare performances

We have now three different analyses for the simulated data, one in terms of FDR and two in terms of fdr. The summary functions indicate that `fdr2d` seems to find the most regulated genes, but this is misleading, as FDR and fdr cannot be compared directly. The function `OCshow` compares the output from multiple analyses graphically, in terms of FDR, by averaging across fdrs. The result in

```
> OCshow(sim1, sim2, sim3, legend=c("FDR","fdr1d","fdr2d"))
```

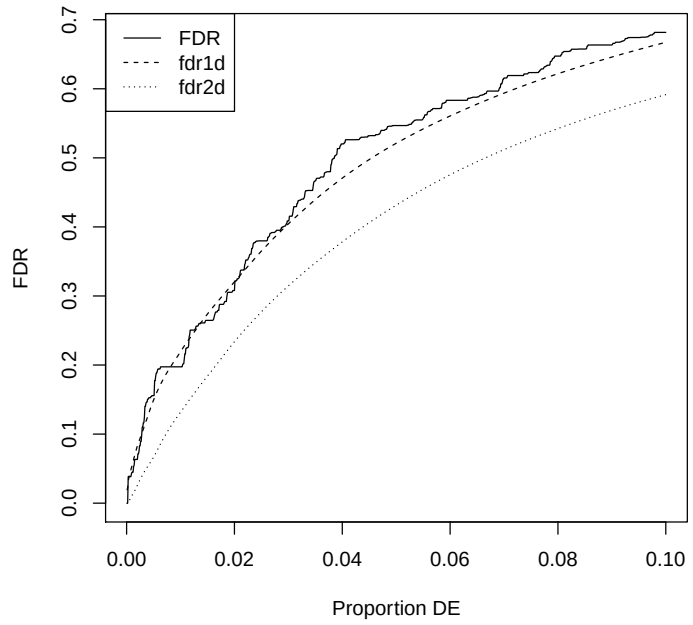

Figure 7: Vertical axis shows the resulting (global) FDR if we declare for each method the proportion of genes with the smallest FDR/fdr shown on the horizontal axis to be differentially expressed.

Figure **??** shows the resulting FDR if we choose the declare the proportion of genes with the smallest FDR/fdr shown on the horizontal axis to be DE. In this case, the original FDR as provided by `EOC` and the FDR based on `fdr1d` are comparable, but the FDR based on `fdr2d` is clearly lower.

5 Estimating the proportion of non-DE genes

An alternative method for estimating the proportion of non-DE expressed genes in a data set is based on fitting a mixture t-distributions to the vector of observed t-statistics, see **?**. In the simplest case, we just compute the t-statistics and specify the number `nq` of mixture components in the call to `tMixture`:

```
> tt = tstatistics(simdat, colnames(simdat))
> tt[1:10,]
```

```
 [1]  1.08042708  0.01011066 -1.07476286 -1.55454200  1.53863046
 [6] -1.52024951 -0.42763371  0.65026341 -0.37170449  0.25691453
```

```
> tm = tMixture(tt, nq=3)
```

In this case, we assume three components, corresponding to down-, up-, and non-regulated genes, and the mixture proportion of the non-regulated genes is the desired estimate:

```
> tm$p0.est
```

```
[1] 0.9151372
```

The estimate is a bit low compared to what we know is true ($p_0 = 0.95$). This is due to the fact that the numerical optimization used by this routine is fairly sensitive to the choice of starting values; it is therefore good practice to vary the starting values for different parameters:

```
> tMixture(tt, nq=3, p0=0.80)$p0.est
```

```
[1] 0.9568759
```

```
> tMixture(tt, nq=3, p0=0.60)$p0.est
```

```
[1] 0.9567759
```

This is essentially the true value for both starting values.

Note that the specification of too many components can lead to spurious mixture components that cannot be distinguished reliably from the non-regulated genes. In order to get reasonable estimates, these components with small non-centrality parameter `delta` are combined with the non-regulated component (which has by definition `delta=0`). E.g.:

```
> tm2 = tMixture(tt, nq=5)
> tm2$p0.est
```

```
[1] 0.9532227
```

```
> tm2$p0.raw
```

```
[1] 0.03364113
```

The estimated proportion `p0.est` is the same as with three components, but it is really the sum of `p0.raw` and the component with non-centrality parameter absolutely smaller than a critical value (0.75 by default):

```
> tm2$p1
```

```
[1] 0.01630578 0.91958160 0.01209428 0.01837721
```

```
> tm2$delta
```

```
[1] -2.587871223 -0.004285759 -1.584812608  2.558282790
```